CHAPTER

1 Introduction to Database Management Systems

1.1 INTRODUCTION

Everyday in our life we encounter with several activities that involves interaction with data or data bases like booking a railway ticket, depositing and withdrawing money from a bank account etc. Initially only manual files were used to keep record of all such data items, but with the growing use of technology the organisations are fast migrating from manual systems to a computerised system, which simplifies the task of recording the data and performing any kind of operation on it.

1.1.1 Data

Data are the raw facts and figures used for describing an attribute about the activity of any object in this world. Raw data refers to the collection of numbers, characters, images, etc. Data is a plural of datum, which means a single piece of information.

Example: In a college Mohinish has a roll no. 109 is data.

1.1.2 Information

Information is a well processed and assembled form of data which is meaningful to the recipients. It is helpful in decision-making and is significant to a business.

Example: Marks obtained by the students in each subject along with their roll no. and names in a mark sheet is information.

1.1.3 Knowledge

Information can be converted to knowledge about historical pattern and future treads. For *example* summary information on retail supermarket sales can be analyzed to provide consumer buying behaviour thus determine which item are most susceptible to promotional effort.

1.1.4 Database

Database is a collection of data that describes the functionality of one or more related organisations.

Example: A college database contains information about – Entities such as students, faculties, courses, etc. and relationship between entities such as student's enrolment in a course.

1.2 DATABASE MANAGEMENT SYSTEM

(KU, Dec 2005)

DBMS is a software system that allows access to data contained in a database and provides an easy and effective method of:

- Defining the information
- Storing the information
- Manipulating the information
- Retrieving the information
- Ensuring the safety of information stored, inspites system crashes or attempts at unauthorised access.
- Avoiding possible anomalous result, if data is shared among several users.

1.3 APPLICATION OF DBMS

1. **School and colleges:** Students, course, fee, information.
2. **Banking:** Customer, accounts, loans, transaction, employee information.
3. **Railway & airlines:** Reservation of tickets and schedule information.
4. **Sales:** Customer, product, purchase, distribution information.
5. **CAD/CAM:** Storing data relating to mechanical, electrical design and production.
6. **On-line retailers:** Online order tracking, maintaining preferences etc.
7. **Human resources:** Information about employee, salaries, etc.
8. **Geographic Information system (GIS):** It stores information such as land management, underwater exploration etc.
9. **Finance:** Information about sales, purchases of stock and bond, managing real time, market data.

1.4 META DATA

(GGSIPU, 2009)

Meta data is defined as the data about data. Consider the contents of a textbook. They summarise topic presented in a textbook so, the content page is like a meta data. Likewise in DBMS, the data about various database objects is collectively stored in a domain and is called metadata.

... of data types, constraints,

1.6 FILE MANAGEMENT SYSTEM (FMS)

File management system stores permanent records in various files and needs different application programmes to extract records and add records to appropriate files.

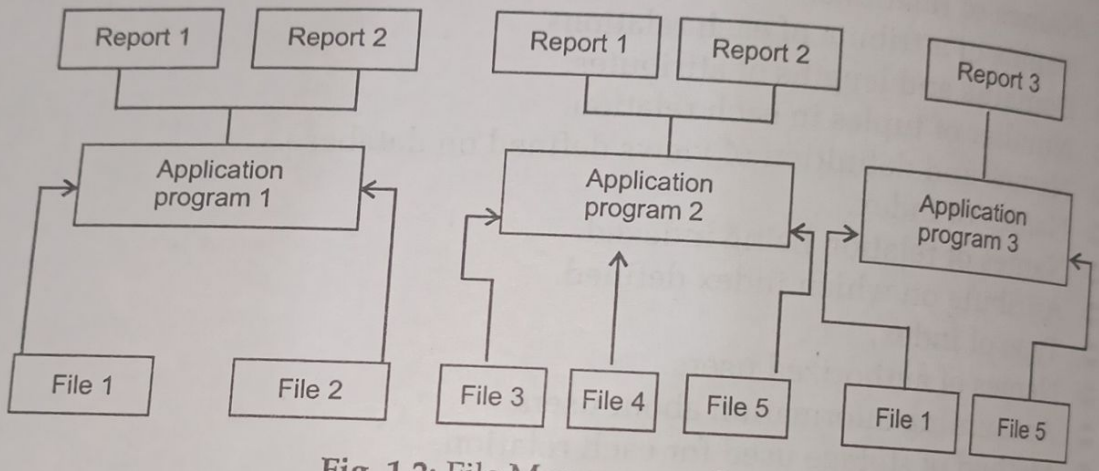


Fig. 1.2: File Management System

For each new file a separate application programme has to be written. There is a tight locking of application programmes and files. If someone decides to modify or add something to the file, all application programmes interacting with that file would need a change.

1.7 DISADVANTAGES OF FMS

(GGSIPU, 2010)

1. Data Redundancy

Since every user or department maintain its own file for data processing so, same information is stored in different files which lead to wastage of storage space.

Example: Student details may be present in a file of personal details as well as accounts file in a college database.

2. Inconsistency

While updating it may happen that an update is applied to one file and not to the other file containing the same information so, the files may become inconsistent, that is various copies of same data may no longer agree.

Example: One user may enter roll no. of a student 205 in one file and other may write the roll no. 110 in the other file for the same student.

3. Integrity Problem

Data stored in a database must satisfies certain types of consistency constraints.

Example: Age of an employee must be in the range between (18-65), these types of constraints are added by writing an appropriate code in application programme. If any new constraints are to be added then these application programmes needs a change.

4. Atomicity Problem

Database should be in consistent state before and after execution of any transaction. If any failure occurs in between, then database must be restored to state prior to start of the transaction.

Example: If a transaction is to transfer Rs.10,000 from an account no. 5678 to account no. 8912, but Rs.10,000 was debeted from account 5678 and not credited to account 8912 due to some failure. It is difficult to ensure these types of atomicity problems in file system.

5. Incompatible File Formats

The structure of a file generated by a C++ programme may be different from that generated by a PASCAL programme. So, this type of incompatibility of files makes them difficult to process jointly.

6. Concurrent Access Anomalies

Same data may be accessed and updated at the same time by multiple users and application programmes which have not been previously coordinated. Supervision may become difficult which leads to inconsistent data.

Example: If one person want to reserve 4 tickets and second person want to reserve 5 tickets simultaneously at the same time for an airline, which has only 5 seats left and if both of them are allocated seats then 9 seats will be reserved but there were only 5 seats left.

7. Difficulty in Accessing Data

While designing the system, every query that can be generated needs to be anticipated by the developer. If any new query is generated by the user, then it would be difficult to access that data.

8. Data Isolation

It is required to determine that which files are needed and how the files are related.

9. Poor Data Control

Since the system was decentralised, it was common for the data field to have multiple names defined by various users, depending on the files it was in. This could result in big confusion.

10. Security Problems

Every user of the system should be able to access only the data they are permitted to see. This is difficult to enforce in file system.

1.8 ADVANTAGES OF DBMS

(KU Dec., 2005; PTU 2011)

In DBMS, files are centralised that is all the files are stored together and there is no duplicacy of data. The user does not even know about the files used by DBMS. They need not also worry about internal details. Only DBA needs to worry about this.

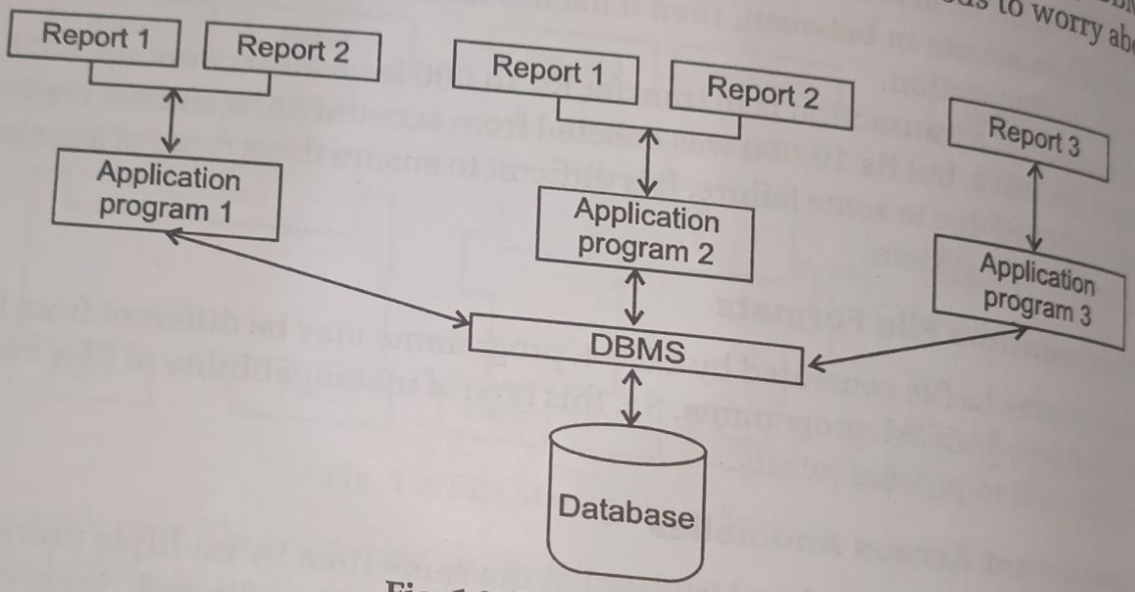


Fig. 1.3: Database System

The advantages of DBMS are:

1. Reduction of Redundancies

By having centralised control of data by DBA, it avoids unnecessary duplication of data and reduces the data storage required.

2. Reduction in Inconsistencies

Since the data is centralised so, only one copy of each data is maintained. Therefore, while updating only that single copy will be modified. Hence, there would be no inconsistency.

3. Shared Data

Existing as well as the new application can share the same centralised data.

4. Improved Data Integrity

Centralised control of data in database system ensures that adequate checks are incorporated in the DBMS to avoid data integrity problem.

5. Atomicity

All database updates between start and end of transaction are atomic i.e. if transaction goes through without problems, it is committed. However, if any problem occurs it is rolled back. This is done by keeping all changes before they start on log files called image records.

6. Data Independence

Users need not be aware of the physical details of the database. They can use the database without knowing how data is stored, size of data etc. They can fully concentrate on logic.

7. Concurrency

DBMS makes sure that two or more users can access the same database at the same time and that the data would remain in correct state. This is achieved by locking when one user is making changes other users are not allowed to even view the data.

8. Security

DBMS makes it easier to enforce security restrictions since data is stored centrally. DBA provides permissions and access right for different users so that unauthorised user cannot see or manipulate the data.

9. Quicker Response

End user can easily extract data, prepare reports without writing any application programme which reduces a lot of programming efforts.

10. Conflict Resolution

DBA resolves the conflicting requirements of various users and applications. He can choose the best file structures or physical representation for the data in storage that gives fast access for most important application while permitting the less important application to continue the database with the relatively slower response.

11. Improved Backup and Recovery

It provides facility for recovering from hardware and software failures, through its backup and recovery system.

1.9 DISADVANTAGES OF DBMS

1. Cost

It involves cost of purchasing and developing the software. It needs hardware upgrading, cost of conversion from old file oriented system to computerised database system, cost of training the manpower to use the new system.

2. Problem Associated with Centralisation

Centralisation increases the security problem and disruption of operation of the organisation because of the down times and failures.

3. Complexity of Backup and Recovery

Due to centralisation there is no duplication of data which means data must be adequately backed up, so that in case of failure data can be recovered.

4. Requirement of New and Specialised Manpower

The organisation needs to maintain specialised manpower to cope up with the rapid change in technology and business needs. For this it needs to train its manpower on frequent basis.

5. Large size of DBMS

DBMS is a large piece of software and occupies many bytes of disk space. It requires substantial amount of main memory to run.

1. Data Definition Language (DDL)

This language is used to specify conceptual schema of a database. Definition of various data elements can be provided by this language such as describing name of the database, table, their attributes, relationships, constraints, authorisation, security, etc.

Example: To create a table student with fields roll_no., name, address, etc. can be done in SQL as

```
Create table student
  {
    Roll_no int,
    Name char [20],
    Address char [50]
  }
```

2. Storage Definition Language (SDL)

These language specify how the database schema is actually stored at physical storage, its implementation details and various access mechanism to use. In short it defines the internal schema of the database.

3. View Definition Language (VDL)

It is used to specify various user views and their mapping to the conceptual schema.

4. Data Manipulation Language (DML)

It enables the user to access or manipulate the data stored in database like retrieval, insertion, modification, deletion of information stored.

Two types of DML are:

(a) ***Procedural DML:*** User need to specify what data is needed and also specifies how to retrieve that data.
Example: **PL/SQL**

(b) ***Non Procedural or Declarative DML:*** User need to specify what data is needed without specifying how to get that data.
Example: **SQL, QBE**

5. Data Control Language (DCL)

These statements control access to data and the database. Various statements that come under DCL are:

Grant — Privilege given to user
Revoke — Take away privilege given to users
Commit — Apply and save workdone by all transactions
Rollback — Restore database to original state since last commit

1.13 SCHEMA, SUBSCHEMA AND INSTANCES

Schema

The overall description or design of the database is called a database schema. Whenever a database for a new organisation has to be created various data element, attributes, relationships are identified i.e. a plan for creating the actual database is formulated. So, this overall plan of the database is known as schema. Data contained in the database may change frequently but the schema remain the same.

Example: A library database with table student, books and issues may have following schema diagram.

Student Roll_No.	Name	Branch	
Book Book_Code	Book Name	Author_Name	No_of_Copies
Issue Roll_No.	Book_Code	Date	

Fig. 1.6: Schema Diagram

Schema can be further divided according to level of abstraction which are—

Logical schema: It describes the schema at logical level i.e. all entities and relationship among them.

Physical scheme: It describes the schema at physical level i.e. how conceptual or logical schema is actually stored on secondary storage.

Subschema

The scheme defined at view level is called a subschema because there may be many schema's at this level due to different user view of data. Subschema is actually a portion of database that is customised for a user or a group of users.

Instance

The collection of data stored in the database at a particular moment is known as instance. Data in the database may change from time to time. So, snapshot of data at a given point of time is instance or state of a database.

Example: Instance of a Student Relation

Student

Roll_No.	Name	Branch
1	Disha	CSE
2	Sonali	CSE

1.14 THREE LEVEL ARCHITECTURE OF DATABASE SYSTEM (DATA ABSTRACTION)

(GGSIPU, 2013, 2011; MDU May, 2009; UPTU 2011-12; PTU, 2011)

Database system provides its users with an abstract view of the data i.e. the system hides certain details of how data are actually stored in the databases so that the user interaction with the database is independent of physical data organisation. Also abstraction is provided to access the same data with different customised view.

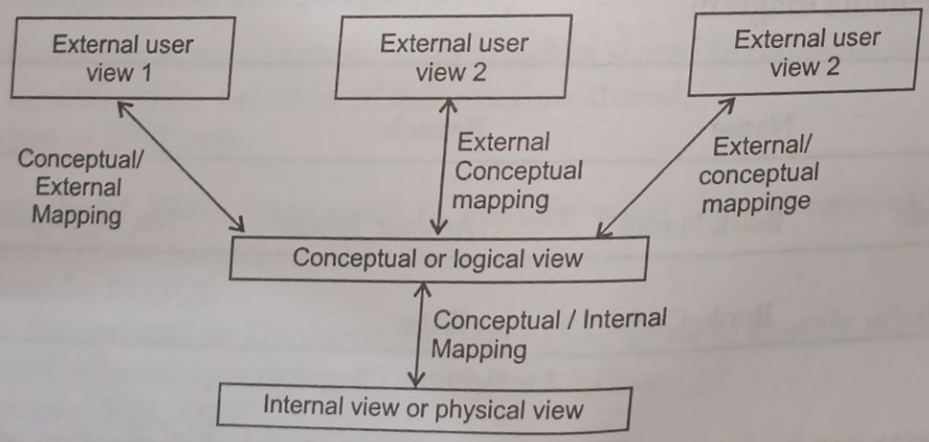
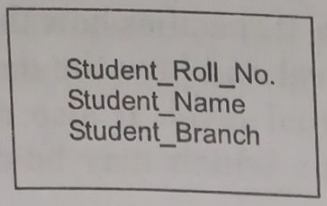


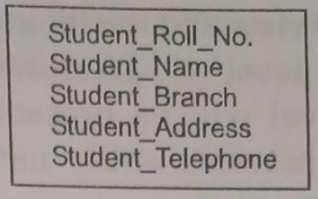
Fig. 1.7: Three-Level Architecture of Database

Different level of abstraction are described below:

1. **View Level:** It is the highest level of database abstraction. This level is concerned with the actual view of the data seen by the user i.e. only those portion of database in which the user is actually interested in or is authorised to see. There may be several view of same data base.

Example: The following diagram show two views of a student database. One user can view only Roll_No., Name and Branch of the student and other can view the address and telephone number in addition to Roll_No, Name and Branch:





2. Logical Level/Conceptual Level: This level gives an single view of entire database. It describes what data entities is stored in the database and what relationship exist among those data entities. It also gives semantic information and constraints on data. All this description is in format independent of it physical representation.

Example:

Student_Roll_No.	:	Integer
Student_Name	:	String
Student_Branch	:	String
Student_Address	:	String
Student_Telephone	:	Integer

3. Physical/Internal Level. It is the lowest level of abstraction and is concerned with how data is actually stored in secondary storage. It also describe various file structure and access methods to be used by database.

Example:

Student_Roll_No.	:	5 dec offset 0 unique
Student_Name	:	String length 25 offset 30
Student_Branch	:	String length 3 offset 33
Student_Address	:	String length 50 offset 83
Student_Telephone	:	10 dec offset 93

Figure shows that student record is of length 93 byte of which first 5 are occupied by Roll No., next 25 by name and so on.

1.15 MAPPING

Each user of the database has his own external view. He can specify any request at this level. This request is transformed by the system and carried to conceptual level. Now, the request at conceptual level is further transformed and carried to internal level. Finally this request is processed and stored in the database. Now, an opposite process starts, the final processed data is again retransformed to satisfy user external view. These transformations discussed above are called mappings.

There are actually two types of mapping.

1. External/Conceptual Mapping: It defines the correspondence among the records and relationship in the external view and conceptual view. Actually it maps names in external view of a user to relevant part of conceptual schema. It tells how to retrieve information from conceptual schema and present it to external view such that it satisfies user external view.

2. Conceptual/Internal Mapping: It defines the correspondence between the

conceptual view and the internal view of database. It specifies how the records at conceptual level will be stored at the internal level and how the data stored at internal level will be presented at the conceptual level. It also resolve any difference between entity names, data types, etc. which may be different in different user view.

Advantages of Three Level Architecture

1. Each user is able to access the same data even through they have different view of data as per their requirements.
2. Any changes made to one user view does not have any effect on other user.
3. A user need not to know anything about physical data storage. If any changes are made to physical storage organisation, then it would not affect the structure of database or user interaction with database.
4. The DBA is able to change the conceptual or internal structure without affecting any user.

1.16 DATA INDEPENDENCE

(GGSIPU, 2009, 2011; MDU May, 2009-2011; PTU 2011, 2009)

It means the ability to use the database without knowing any representation details. It also states that if any changes are made at one level of database system, then it would not affect any higher level of database system.

Two types of data independence are:

1. **Physical Data Independence:** It means any changes made to the internal schema (such as using new storage devices, changes in access method, using different file organisation etc.) will not affect the conceptual schema. To absorb these changes only mapping need to be adjusted and no new application programme needs to be written.
2. **Logical Data Independence:** It means any changes made to logical schema (such as addition, deletion of entities, relationship etc.) will not affect the external schema. Only view definition and mapping need to be changed and no new application programme needs to be written

1.17 DATABASE USERS

(PTU 2009)

Database users can be categorised depending upon their level of interaction with DBMS.

Naïve Users

They interact with the database through a form interface by filling in particular field being displayed on the screen. This invokes an appropriate application programme which process the instructions given by the users and then generate the required response.

adjustments.

1.18 DATABASE MODELS

(MDU May, 2009)

A database or data model can be described as a collection of various tools for describing data, relationship existing among different data items, data semantics and various constraints to be applied on that data. A data model describes the structure of a database and provides means to achieve abstraction i.e. how to define data at physical, logical or view level. Apart from structure of data, a data model also define a set of operations that can be performed on the data. A database model is categorized into three type's namely conceptual data model, representational data model and physical data model.

Conceptual Data Model

This daia model describes the information of an organization in a way that is independent of any implementation level details. An example of conceptual data model is entity-relationship model which consists of collection of object called entities and relationship that exist among those entities. In the figure below Teacher and Student are entities and Teach models relationship that exist among the two entities.

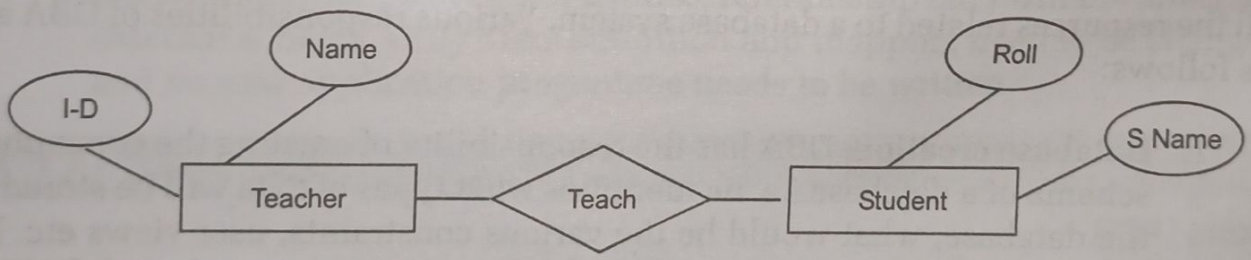


Fig. 1.8

Representational Data Model

These data model hide some data storage details from users and can be directly implemented on computer system. Some of the representational data model are described here.

Hierarchical Data Model

Hierarchical DBMS were popular from late 1960's. This database was developed by IBM for its IMS (Information Management System) database. It is one of the oldest DBMS. Data in hierarchal data model is organized as a tree structure. There is a hierarchy of parent child relationship between data segments. This implies that there may be repeating information in child data segment. In this data model there is a root data segment, below which there is one or more child data segment, each of which may have its one or more child segments and so on. Hence this data model can model only one-to-many relationships. In this model data is organized as series of data records which have one or more field called segments which form the node of a tree. Instances of a specific record are grouped together as a record. An example of hierarchical model is shown below

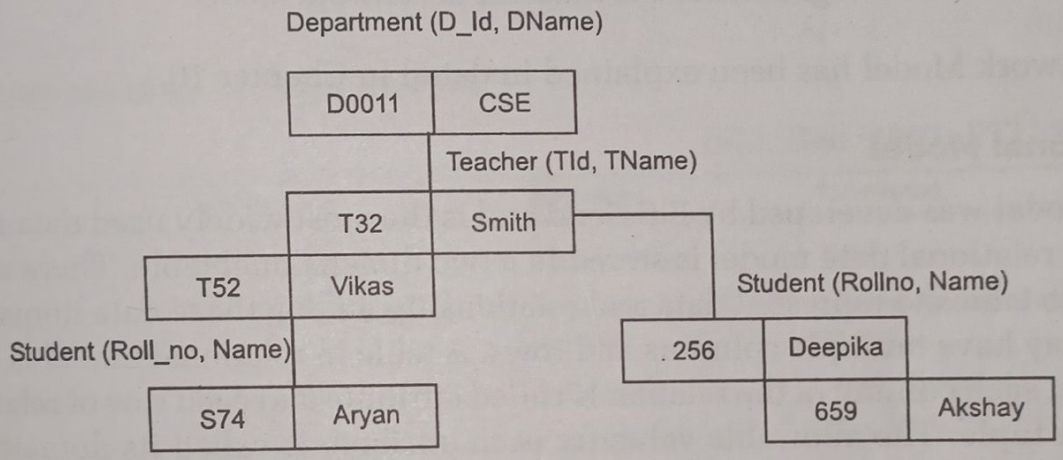


Fig. 1.9: Sample Database of Hierarchical Model

The main advantage of this data model is high speed of access and ease of update due to its predictable structure. The limitation of this model is that it does not support flexible data access because element can be accessed by following a particular path. Moreover links are hardcoded as data structure, so link cannot be modified. Hierarchical model has been explained in detail in chapter 10.

Network Data Model

(GGSIPU, 2008)

A Network data model is an extension of hierarchical data model. This model was formalized in 1971 by Database Task Group (DBTG) in the conference on Data System Language. This model uses directed acyclic graphs with nodes and edges where nodes represent entity sets and edge represents relationship between entity sets. Data in network model is modelled through a set construct. A set consist of an owner record type, a set name and a member record type. A member record type can have that role in more than one record type. So a network model can model one-to-many and many-to-many relationship as a child can have more than one parent. An example of Network model is shown below:

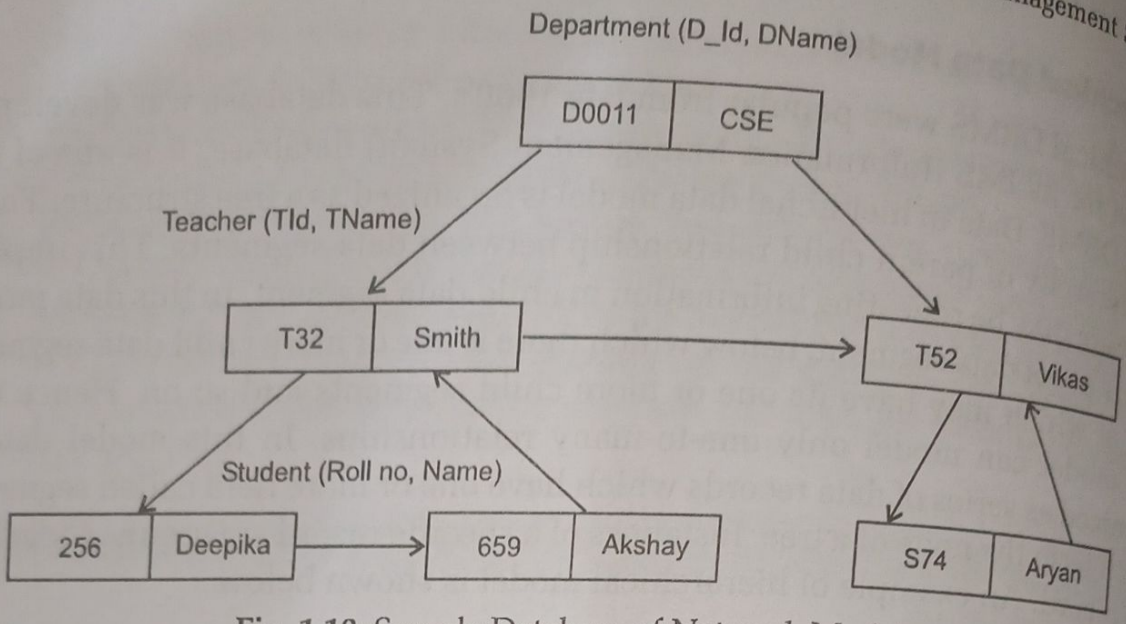


Fig. 1.10: Sample Database of Network Model

Network Model has been explained in detail in Chapter 10.

Relational Model

This model was developed by E.F Codd and is the most widely used data model. Data in relational data model is stored in a two dimensional table. There may be multiple tables to represent data and relationship among those data items. Each table may have multiple columns and rows. A table in relational model is called relation, each column of the relation is called attribute and each row of relation is called a tuple. The allowable value for each attribute is called its domain. The relationship between two relations is formed by a common attribute and not any physical link. An example of relational model is shown below:

Department

Did	DName
D0011	CSE

Teacher

TId	TName	DId
T32	Smith	D001
T52	Vikas	D00l

Student

Rollno	SName	TID
256	Deepika	T32
574	Aryan	T52
654	Akshay	T32

Fig. 1.11: Sample Database of Relational Model

Advantages of relational database are:

Structural Data Independence: Changes made in database structure does not affect DBMS capability to access data.

Simplicity: Allows designers to focus on logical view rather than physical storage details.

Adhoc query Capability: Powerful, flexible nad easy to use query language.

Note: Hierarchical, Network and Relational data model are also called record based data model as data is represented in form of records.

Object Relational Data Model

Relational technology is not good enough at handling the needs of complex information systems as they stores simple data items such as integer, character etc. Moreover operations that could be performed over them were simple and predefined. Object relational database system adds additional functionally to relational database system such as objects, classes and inheritance in database schemas and query language. These new facilities integrate management of traditional fielded data, complex objects such and diverse binary media such as audio, video, images, and applets, time-series and geospatial data.

Physical Data Model

This data model describes data in terms of collection of files, record format, access path etc. It specifies how the database would be executed on a particular DBMS software.

1.18.1 Comparison of Data Models

(KU, Dec. 2005; PTU 2009)

Item	Hierarchical	Network	Relational
Data Structure	File or records	File, Records	Table
Data Access	Navigational and easy	Navigational and difficult	Non-navigational and easy
Access Language	Procedural	Procedural	Non-procedural
Identity	Record based	Record based	Value based
Relationship	Logical proximity in a tree	Intersecting network	Attribute of one table exit in other table
Data Integrity	No	No	Part of model
Query language	COBOL	COBOL	SQL
Modification	Difficult	Difficult	Easy
Structural Independence	No	No	Yes
Consistency	Due to information replication inconsistency occurs	More data consistency then hierarchical	More data consistency than other
User required to know	Structure of database	Structure of database	Tables only

1.20 CLIENT SERVER ARCHITECTURE OF DATABASE SYSTEM

(GGSIPU 2007; MDU May, 2010, 2011)

A client server architecture consist of a server and a set of clients. DBMS resides on server and perform all functions related to database management like data definition, manipulation, integrity, security, etc. A client is a personal computer or a workstation that run various DBMS applications. These servers and client computers are connected to a network. Client request for a service by running an application programme. The server response to the request by returning results to the clients.

A client server architecture may be divided into:

1. **Two-tier client server architecture:** In this type of architecture, the client invokes database system functionality at server machine through query language. ODBC, JDBC are used for interaction between the client and server.

   Advantage of two-tier architecture is that they are simplest to implement and performs well with moderate number of client. Drawback is that software maintenance can be difficult because PC clients contains a mixture of presentation, validation and business logic code. To make any changes, code must be modified on PC clients. Moreover it gives poor performance when large number of clients submit their requests.

   (GGSIPU, 2013)

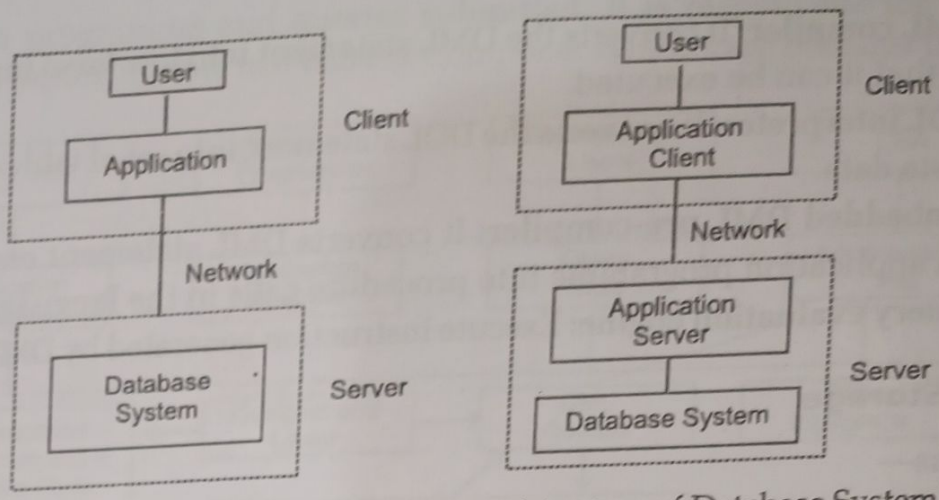


Fig. 1.13: Two-Tier and Three-Tier Architecture of Database System

2. **Three-tier client server architecture:** Client machine acts as a front end and communicates with the application server, through a form interface. The application server in turn communicates with database system to access data.

Advantages of Client-Server Architecture

1. All data is stored on servers, which generally have far more better security then clients.
2. It function with multiple different clients of different capabilities.
3. It increases overall performance of DBMS.
4. Provides better user interface.
5. Since data is centralized, updates to that data is easy.
6. It reduced cost.

Disadvantages of Client-Server Architecture

1. If Server fails, whole networks fails.
2. As number of simultaneous client requests to a given server increases, the server can become overloaded.
3. If server crashes, there can be loss of data.
4. Network may be error prone.